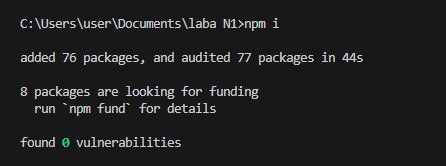
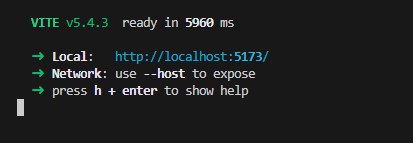
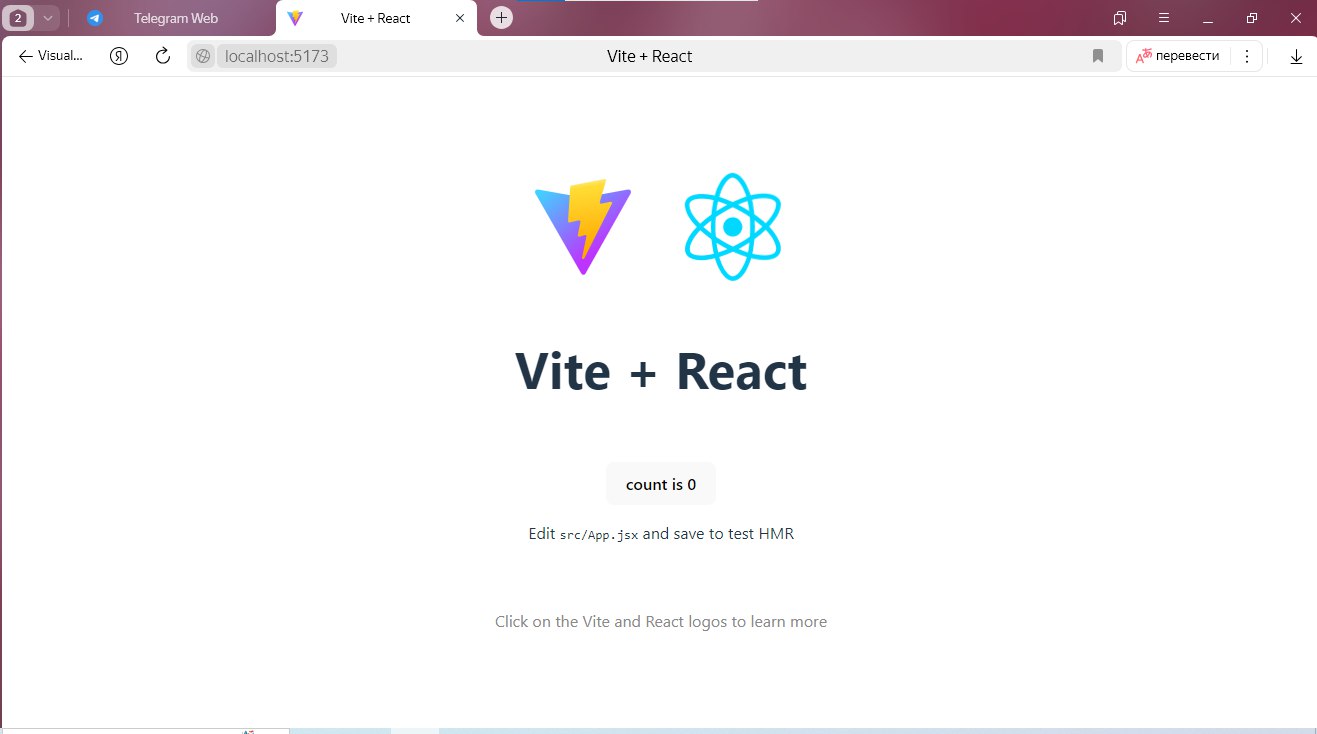
**ЛАБОРАТОРНАЯ РАБОТА N1. 4 КУРС**

*Выполнила:* Морозова Милана – ИСП-421р

***1. Установка***







*Задание №1*

Автор следующего кода хочет вернуть три инпута:

function App() { return <input><input><input>;}

Код, однако, не работает. Исправьте ошибку автора кода.

Решение:

function App() {

  return (

    <>

      <input></input>

      <input></input>

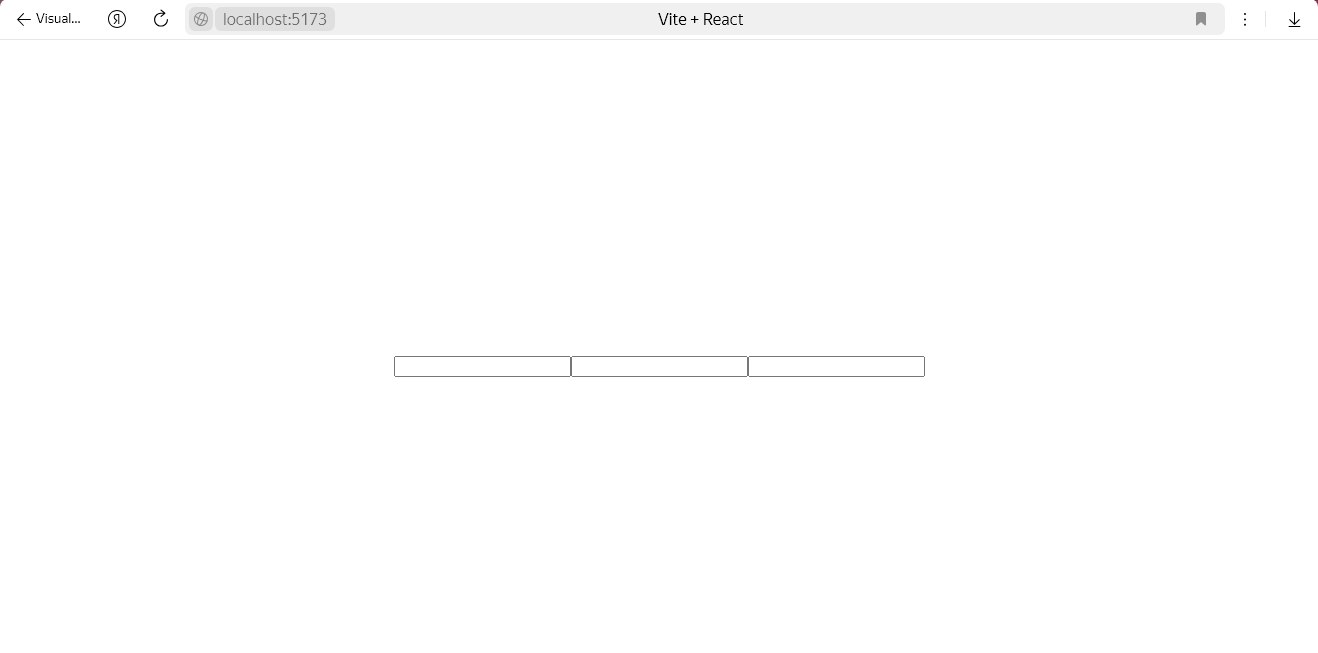
      <input></input>

    </>

  )

};

Итог:



*Задание №2*

Дан следующий код:

function App() {

const str1 = "text1";

const str2 = "text1";

return (

<div>

<p></p> <p></p>

</div>

);

}

Вставьте первую константу в первый абзац, а вторую константу - во второй.

Решение:

function App() {

  const str1 = "text1";

  const str2 = "text2";

  return (

    <div>

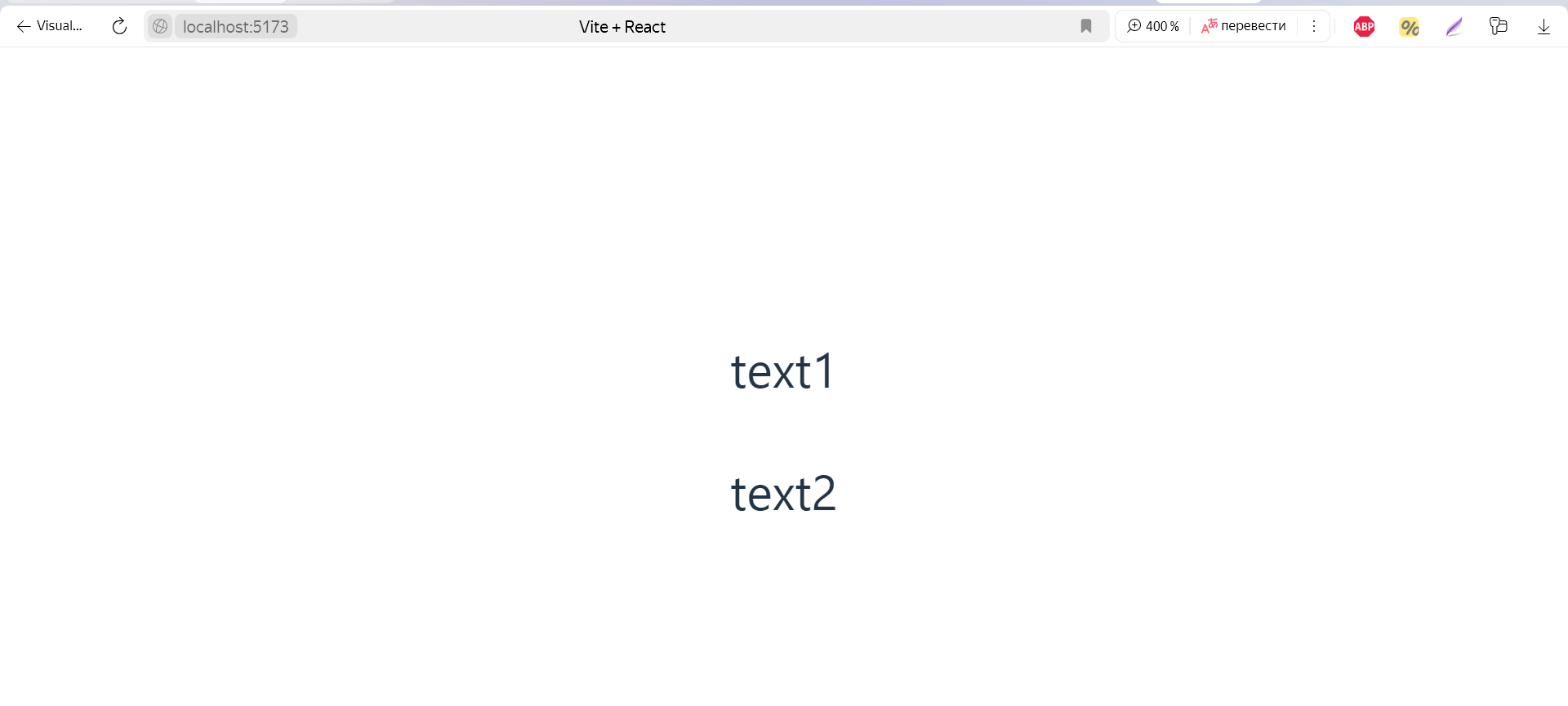
      <p>{str1}</p> <p>{str2}</p>

    </div>

  );

}

Итог:



*Задание №3*

Дан объект:

function App() {

const obj = { name: "john", surname: "smit" };

}

Сделайте так, чтобы результатом рендеринга был следующий код:

<p> name: <span>john</span>, <br> surname: <span>smit</span></p>

Для значений имени и фамилии используйте значения элементов объекта.

Решение:

function App() {

  const obj = { name: "Milana", surname: "Morozova" };

  return (

    <p>

      name: <span>{obj.name}</span>, <br />

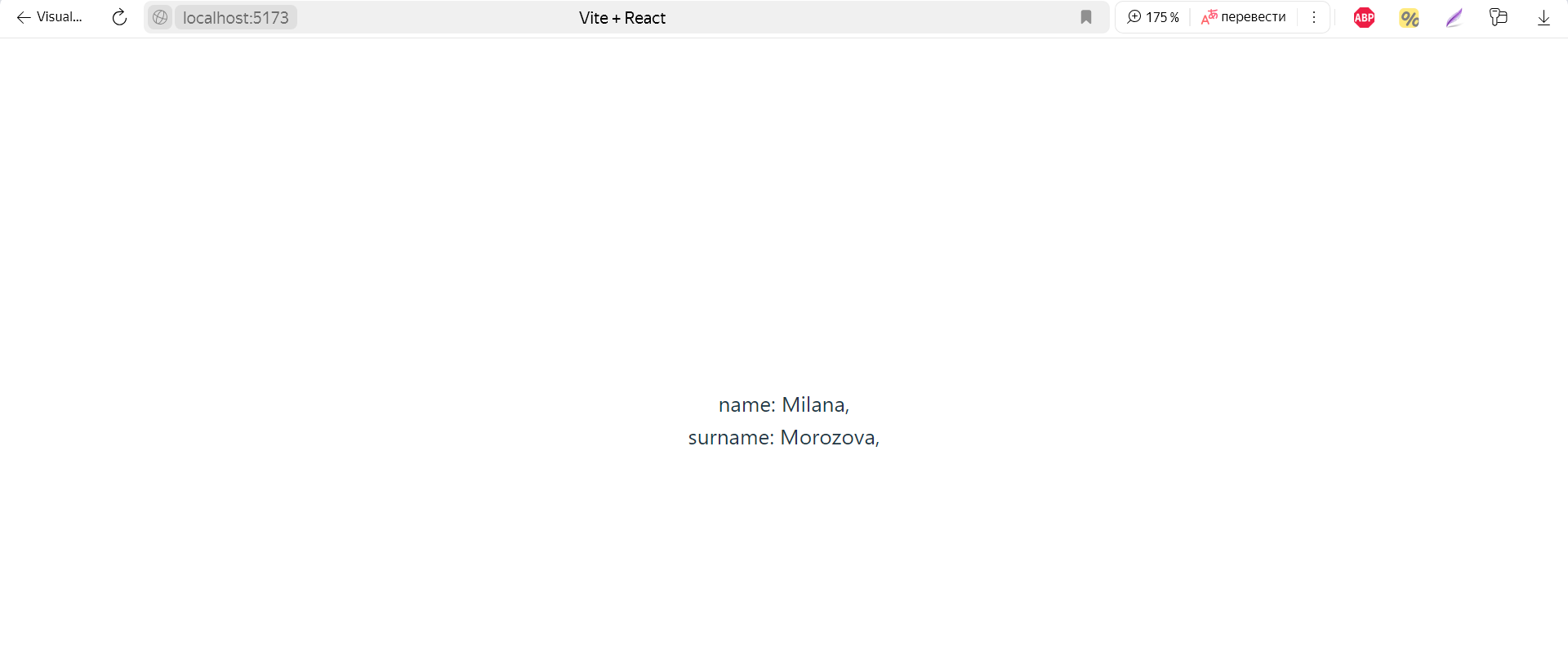
      surname: <span>{obj.surname}</span>,

    </p>

  );

}

Итог:



*Задание №4*

Сделайте функцию getDigitsSum, которая будет находить сумму цифр переданного

числа. С ее помощью выведите на экран сумму цифр числа 123.

Решение:

function App() {

  function getDigitsSum(number) {

    let sum = 0;

    let numStr = number.toString();

    for (let i = 0; i < numStr.length; i++) {

      sum += parseInt(numStr[i]);

    }

    return sum;

  }

  const sum = getDigitsSum(123);

  return (

    <div>

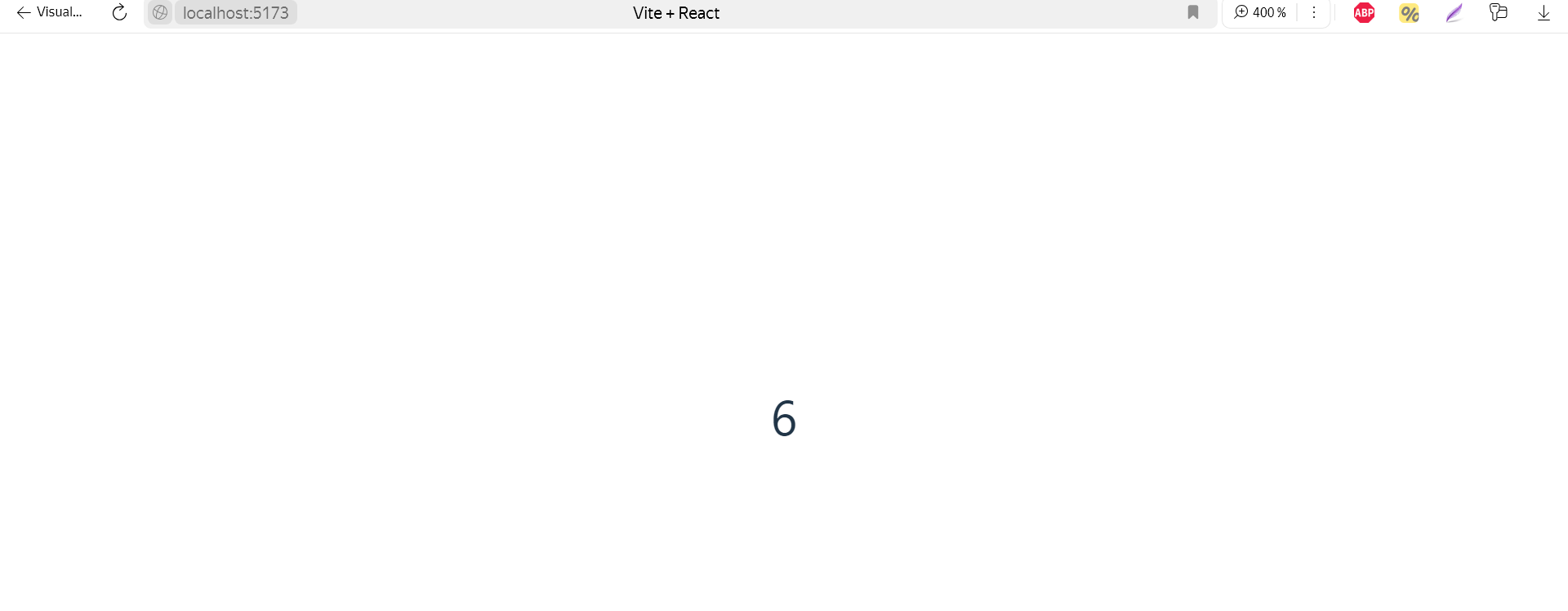
      {sum}

    </div>

  );

}

Итог:



*Задание №5*

Дан следующий код:

function App() {

function show1() {

alert(1);

}

function show2() {

alert(2);

}

return <div>

<button>act1</button>

<button>act2</button>

</div>;

}

Сделайте так, чтобы по клику на первую кнопку срабатывала первая функция, а по

клику на вторую кнопку - вторая функция.

Решение:

function App() {

  function show1() {

    alert(1);

  }

  function show2() {

    alert(2);

  }

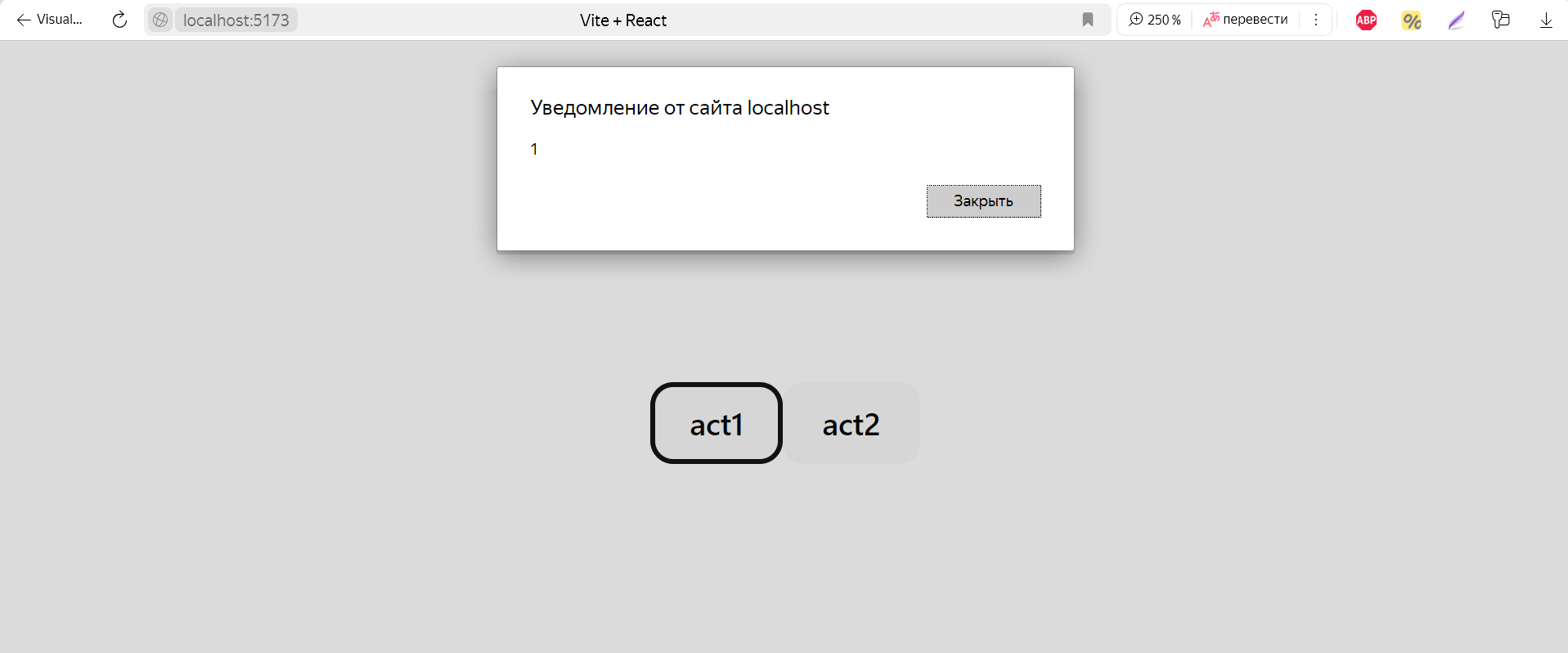
  return <div>

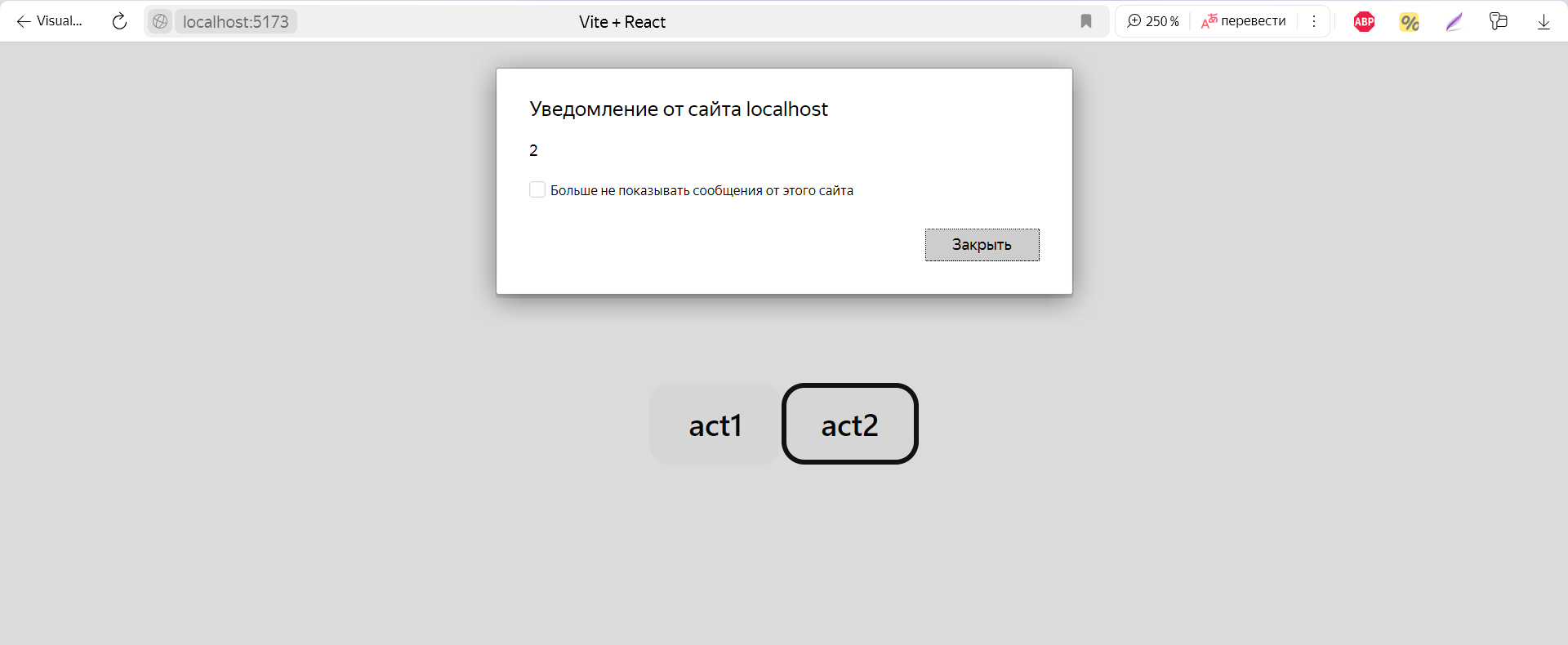
    <button onClick={show1}>act1</button>

    <button onClick={show2}>act2</button>

  </div>;}

Итог:





*Задание №6*

Даны три кнопки:

function App() {

return <div>

<button>act1</button>

<button>act2</button>

<button>act3</button>

</div>;

}

Сделайте так, чтобы по клику на первую кнопку алертом выводилось число 1, по

клику на вторую кнопку - число 2, а по клику на третью - число 3.

Решение:

function App() {

  function showMess(text) {

    alert(text);

  }

  return <div>

    <button onClick={() => showMess('1')}>act1</button>

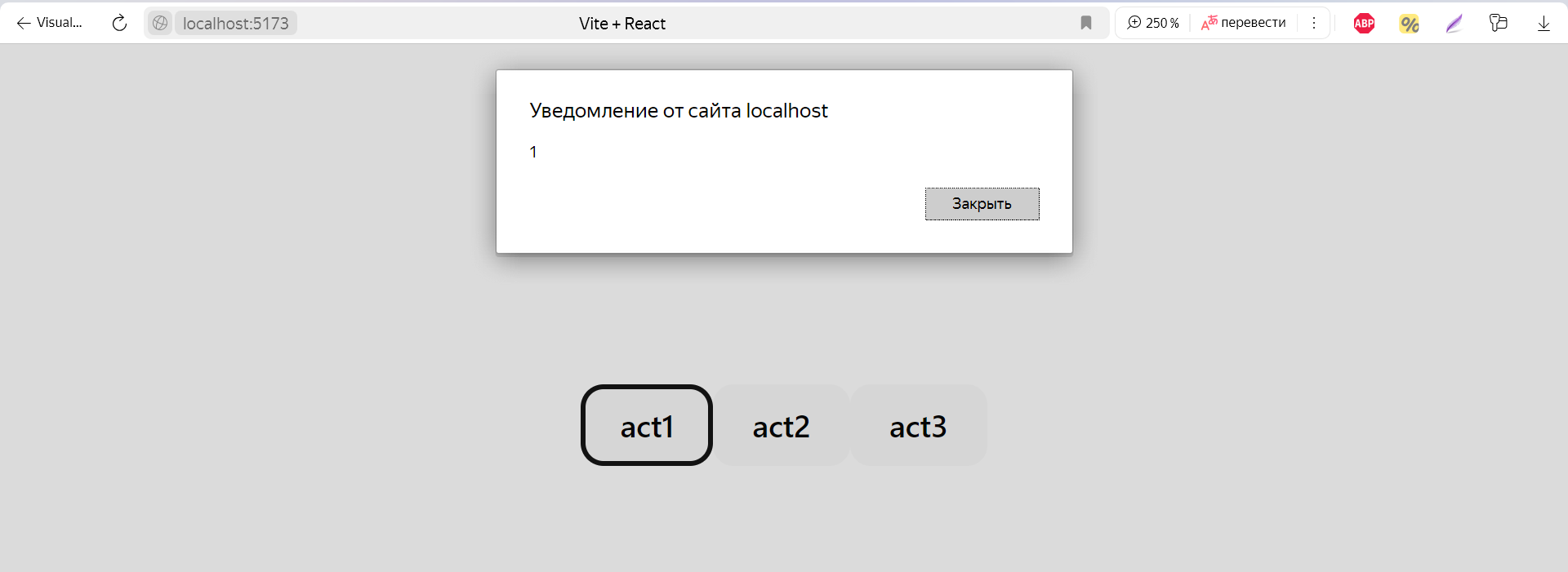
    <button onClick={() => showMess('2')}>act2</button>

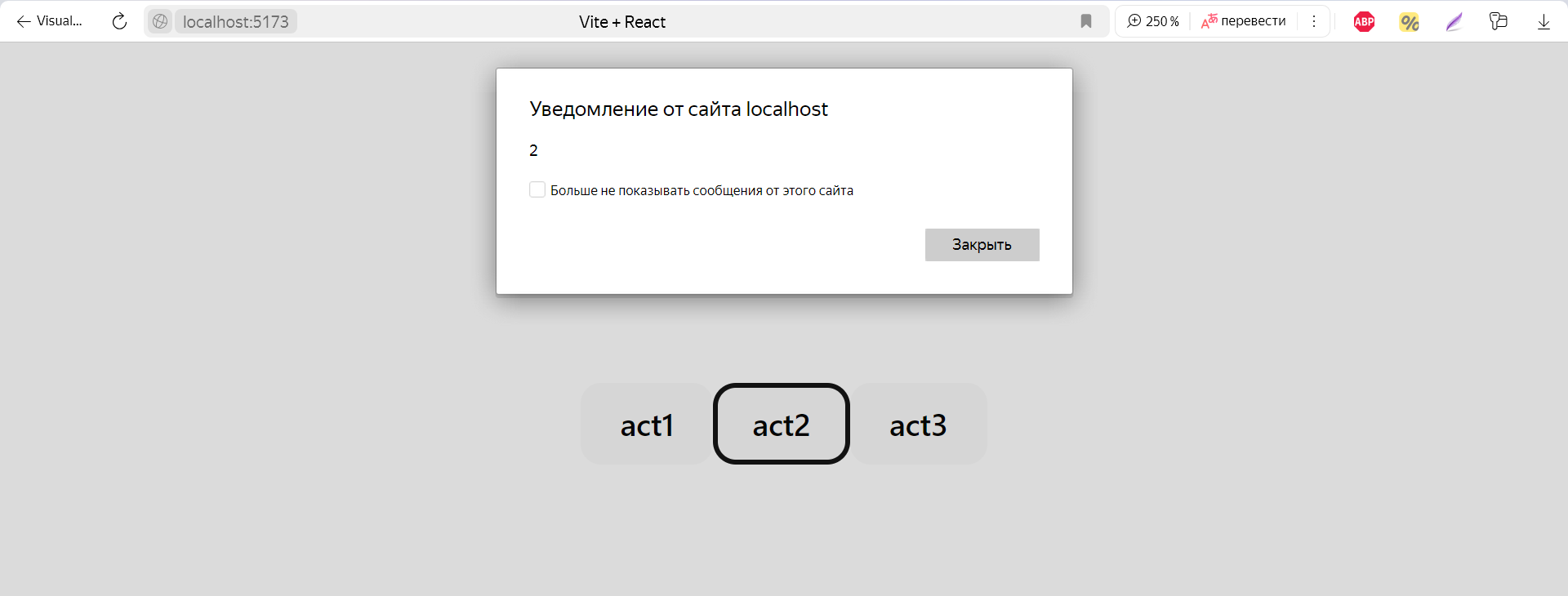
    <button onClick={() => showMess('3')}>act3</button>

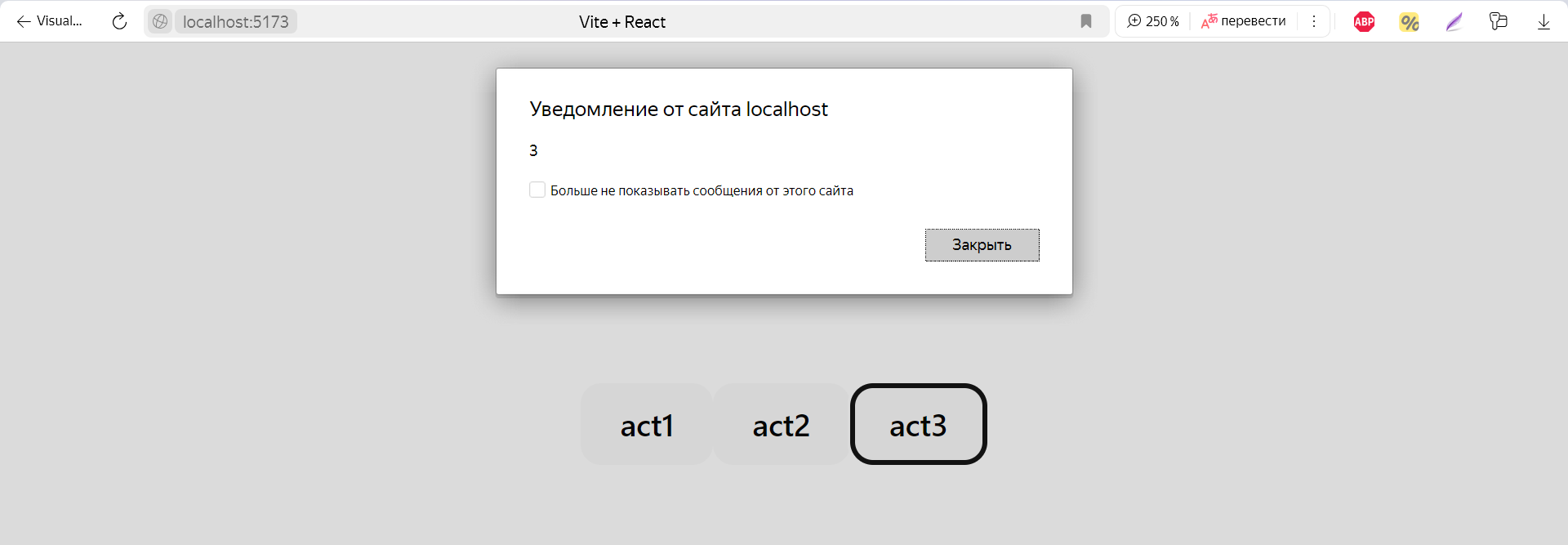
  </div>;

}

Итог:







*Задание №7*

В компоненте App дан следующий массив:

const users = [

{ name: "user1", surn: "surn1", age: 30 },

{ name: "user2", surn: "surn2", age: 31 },

{ name: "user3", surn: "surn3", age: 32 },

];

Выведите элементы этого массива в виде списка ul.

Решение:

function App() {

  const users = [

    { name: "user1", surn: "surn1", age: 30 },

    { name: "user2", surn: "surn2", age: 31 },

    { name: "user3", surn: "surn3", age: 32 },

  ];

  return (

    <div>

      <ul>

        {users.map((user) => (

          <li key={user.name}>

            {user.name} {user.surn} ({user.age})

          </li>

        ))}

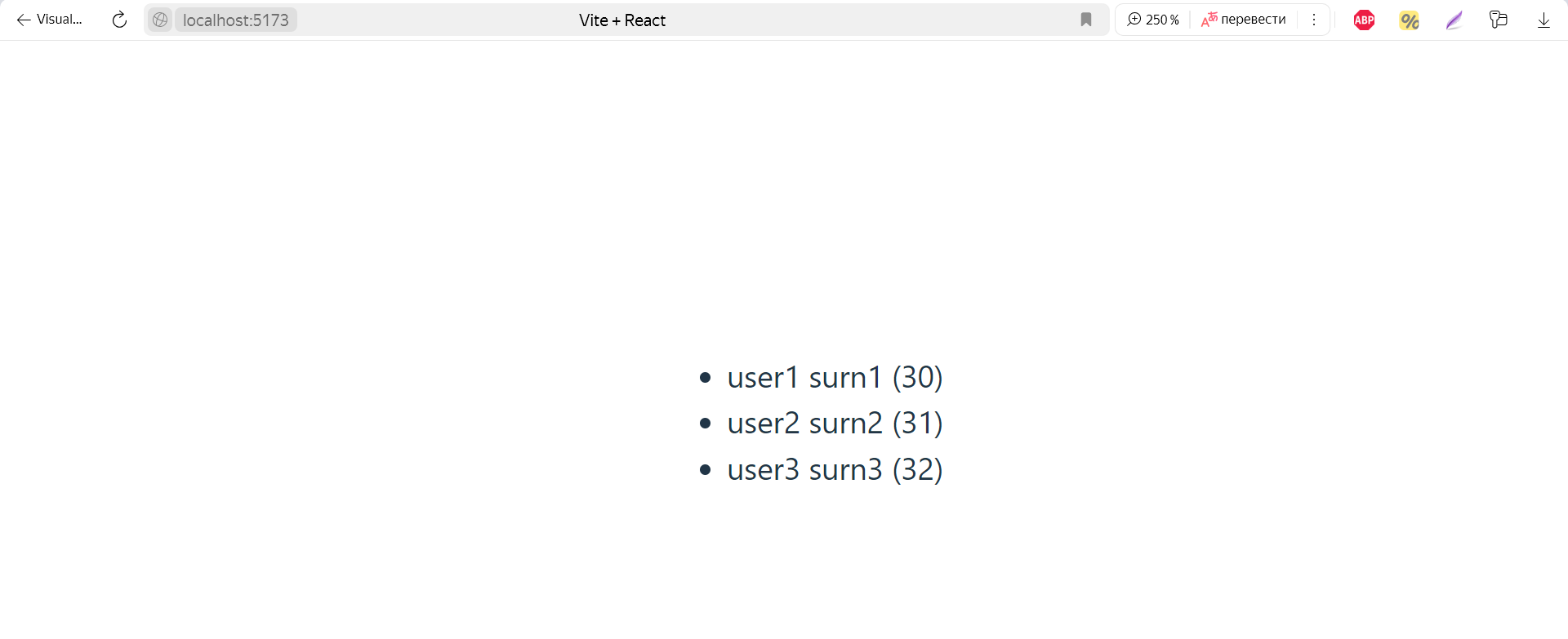
      </ul>

    </div>

  );

}

Итог:



*Задание №8*

Пусть вы хотите отображать на экране данные юзера: его имя, фамилию, возраст.

Сделайте для этого соответствующие стейты с начальными значениями.

Решение:

function App() {

  const [name, setName] = useState("Milana");

  const [surname, setSurame] = useState("Morozova");

  const [age, setAge] = useState("18");

  return (

    <div>

      <span>{name}</span> <br />

      <span>{surname}</span> <br />

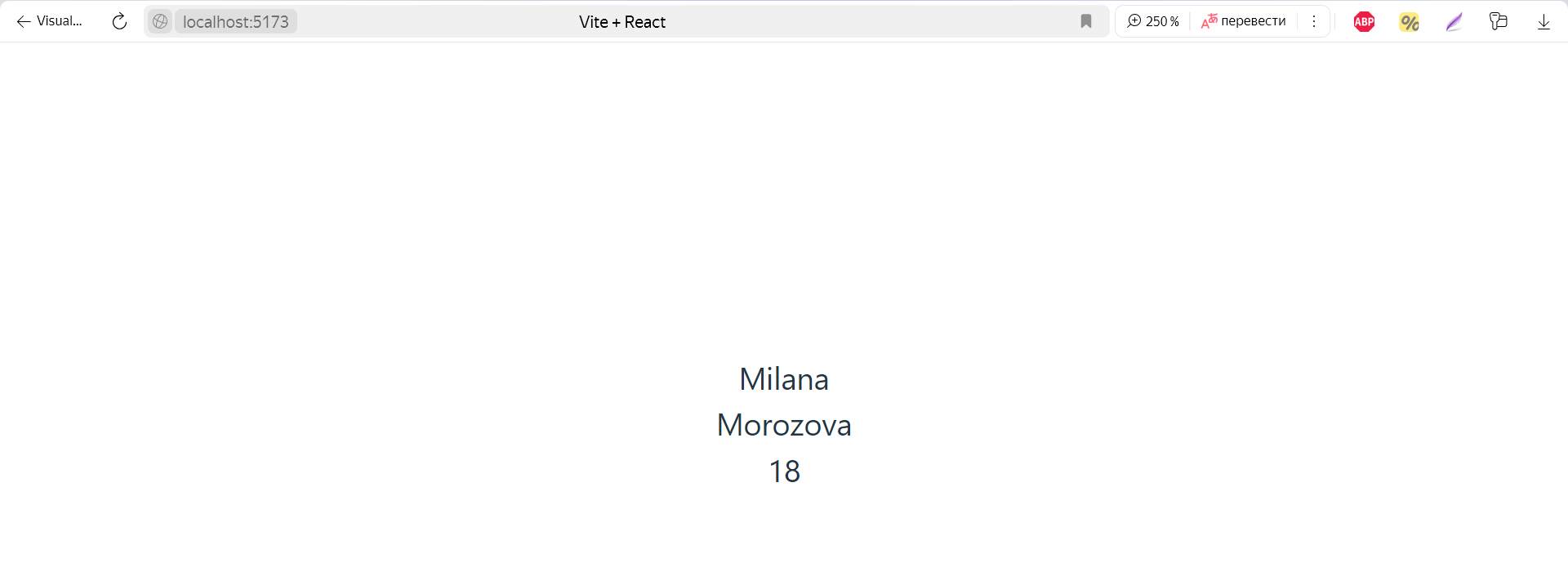
      <span>{age}</span>

    </div>

  );

}

Итог:



*Задание №9*

Сделайте кнопку, нажатие на которую будет банить пользователя и кнопку, нажатие на

которую будет разбанивать пользователя.

Решение:

import './App.css';

import React, { useState } from "react";

function App() {

  const [isBanned, setIsBanned] = useState(false);

  const handleBan = () => {

    setIsBanned(true);

  };

  const handleUnban = () => {

    setIsBanned(false);

  };

  return (

    <div>

      <span>

        {isBanned ? "Забанен" : "User"}

      </span> <br />

      <button onClick={handleBan} disabled={isBanned}>Забанить</button>

      <button onClick={handleUnban} disabled={!isBanned}>Разбанить</button>

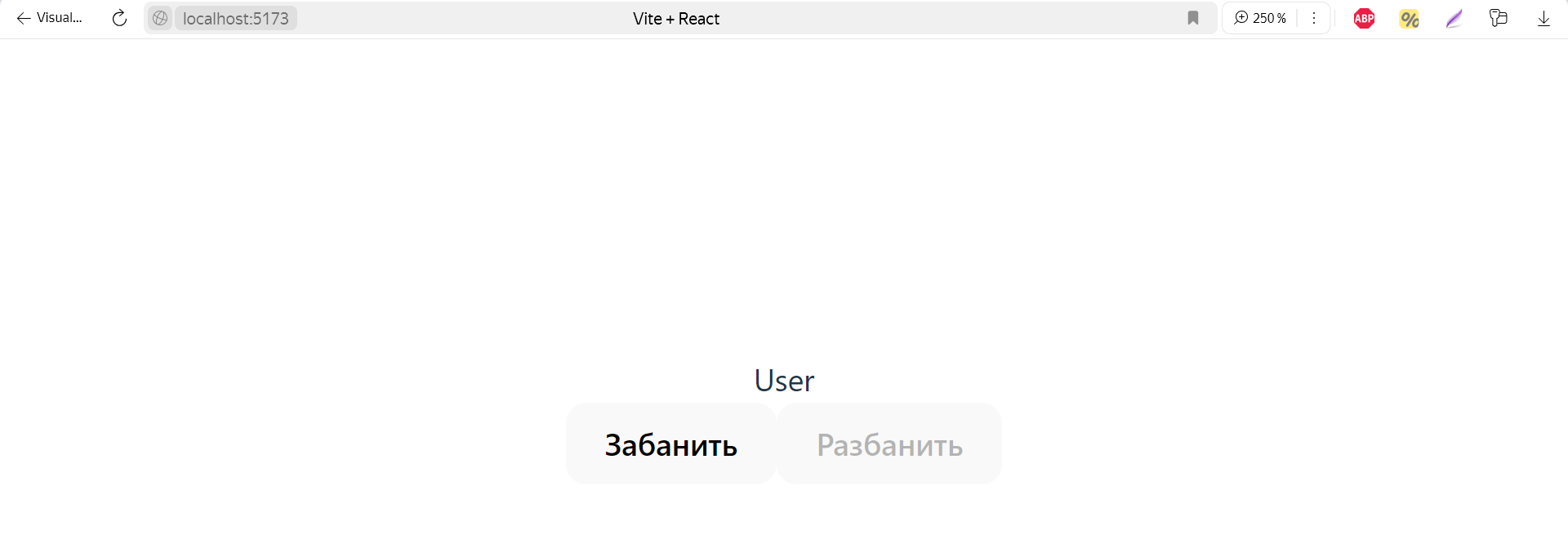
    </div>

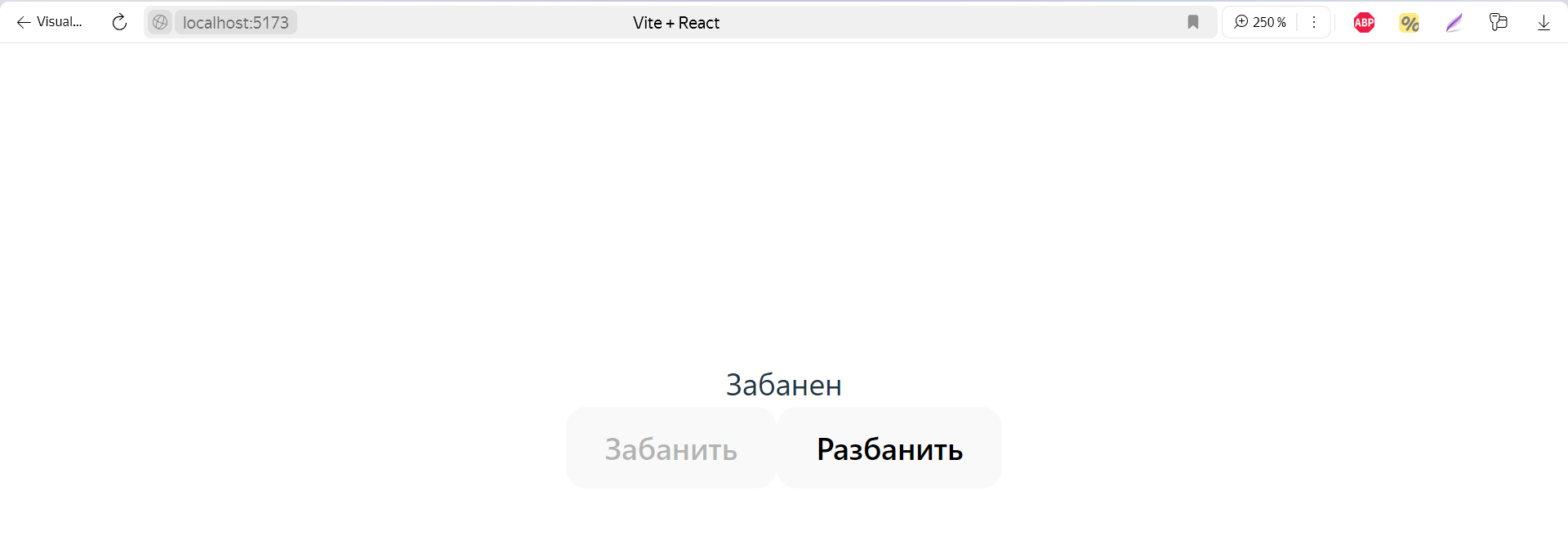
  );

}

export default App;

Итог:





*Задание №10*

Модифицируйте предыдущую задачу так, чтобы из двух кнопок всегда была видна только

соответствующая. То есть, если пользователь забанен, то видна кнопка для разбанивания,

а если не забанен - для забанивания.

Решение:

import './App.css';

import React, { useState } from "react";

function App() {

  const [isBanned, setIsBanned] = useState(false);

  const handleBan = () => {

    setIsBanned(true);

  };

  const handleUnban = () => {

    setIsBanned(false);

  };

  return (

    <div>

      <span>

        {isBanned ? "Забанен" : "User"}

      </span> <br />

      {isBanned ? (

        <button onClick={handleUnban}>Разбанить</button>

      ) : (

        <button onClick={handleBan}>Забанить</button>

      )}

    </div>

  );

}

export default App;

Итог:

